Imitation of concept learning by honey bees using Vector Symbolic Architectures

Denis Kleyko[a,*], Evgeny Osipov[a], Ross W. Gayler[b], Asad I. Khan[c], Adrian G. Dyer[d]

[a]*Department of Computer Science Electrical and Space Engineering, Luleå University of Technology, 971 87 Luleå, Sweden*
[b]*Independent Researcher, Melbourne, VIC, Australia*
[c]*Clayton School of Information Technology, Monash University, Clayton, VIC, Australia*
[d]*Media and Communication School, Royal Melbourne Institute of Technology, Melbourne, VIC, Australia*

Abstract

This article presents a proof-of-concept validation of the applicability of Vector Symbolic Architectures to form a central part of an online learning architecture. It is demonstrated that Vector Symbolic Architectures make feasible the structured combination of features/relations that have been detected by a perceptual circuitry and applying those relations to novel structures without requiring the massive training that is needed for classical neural networks that depend on trainable connections.

The system is showcased through functional imitation of the concept learning by honey bees. The presented work uses the results of the real world experiment with honey bees Avarguès-Weber et al. (2012) for benchmarking. It is demonstrated that the proposed pipeline features similar learning curve and the accuracy of generalization as in the living bees. The main claim of this article is that there is a class of simple artificial systems, which do not require implementation of computationally intensive cognitive architectures. Thus in certain applications rather advanced cognitive behavior could be implemented using simple techniques.

Keywords: Vector Symbolic Architecture, hyperdimensional computing, distributed data representation, concept learning, cognition, inference.

1. Introduction

In the past two decades the honeybee has become established as an important model insect for insights into machine vision (Srinivasan (2006), Srinivasan (2010), Srinivasan (2011)), and recent advances in understanding cognition in a bee brain potentially provides a useful model for how machine vision may efficiently learn. For example,

*Corresponding author

Email addresses: denis.kleyko@ltu.se (Denis Kleyko), evgeny.osipov@ltu.se (Evgeny Osipov), r.gayler@gmail.com (Ross W. Gayler), asad.khan@monash.edu (Asad I. Khan), adrian.dyer@rmit.edu.au (Adrian G. Dyer)

behavioral experiments on free flying honeybees reveal that these insects estimate distance traveled by measuring the optic flow of spatial information content (Srinivasan (2010)), a principle that has been incorporated in flying machines (Srinivasan (2011), Soccol et al. (2007)). This article presents the design of an artificial learning system, which imitates the concept learning in a living organism. The approach is inspired by the experiments with honey bees, which demonstrate their capabilities of learning new concepts Giurfa et al. (2001), Avarguès-Weber et al. (2012), Avarguès-Weber et al. (2014). In short the experiment that was used to understand new insights into artificial systems is about training bees to select one or another direction in a maze illustrated in Figure 3. This impressive learning is based on the visual input (a certain pattern of two 2D geometrical shapes with specific attributes arranged in a certain order on a plane) indicating the direction towards a reward (sucrose solution), the other direction resulted in a penalty (quinine solution). The behavioral experiments were designed to use the reinforcement learning in the training phase.

Amongst the main challenges for the computer imitation of the learning behavior of the bees tested in Avarguès-Weber et al. (2012) are the learning a concept on a very small data set (30 in the behavioral experiments) and a capability of incremental update the learned concept allowing for flexibility in interpretation of previously unseen visual targets.

1.1. Contribution of the article

This article focuses on the proof-of-concept validation of the applicability of Vector Symbolic Architectures (VSA) for imitating the learning behavior of the living organisms. The proposed artificial online concept learning system adopts Vector Symbolic Architectures. The contribution of the article is the first time usage of Vector Symbolic Architecture for functional modeling a part of the concept learning by honey bees. That is the proposed architecture approximates the functional behavior of bees without mapping it to bee-specific mechanisms/anatomy. It was shown in Eliasmith (2013) that VSAs are relatively easy to implement neurally, therefore it would be worthwhile in the future to attempt to find the fingerprints of VSA-based processing in bee behavior.

The biological inspiration of the work presented in this article is three-fold. Firstly, the main practical aim is to model a part of the cognitive behavior of a biological organism, i.e. experimentally demonstrated concept learning in honey bees. Secondly, the technique used to model this behavior, i.e. distributed representations and vector symbolic architectures are both bio-inspired in a way that they model activities of neurons in human brain when representing and reasoning upon structured knowledge. In particular Kanerva's learning by example approach to VSA reasoning is adopted for the simplest possible implementation Kanerva (2000). Finally, the online concept learning architecture is anchored to psychologically-plausible model for similarity analysis, i.e. the hybrid model of structural, spatial and featural similarity. It is demonstrated that the proposed system resembles the characteristics of the bees learning system that is: it requires small number of training trials; it is able to continuously update the learned concept; it is flexible in interpretation of the previously unseen visual stimuli.

The main achievements of the article are summarized as:

- Vector symbolic architectures was used to represent the structured combination of features/relations and applying those relations to novel structures in the online learning context;
- The choice of the relation represesentation is motivated by the classical analogical similarity models;
- VSA representation of relations forms a central part of the learning pipeline with a perceptual circuitry and makes feasible applying those relations to novel structures without requiring the massive training that is needed for classical neural networks;
- A proof-of-concept validation of the proposed online learning pipeline showcased on the example of the discovered concept learning capabilities by honey bees.

1.2. Delimitations

While aiming at presenting a generic architecture for online concept learning certain design choices were tailored to the specifics of the considered biological experiments. That is the system is designed to analyze simple two dimensional visual stimuli consisting of a pair of objects as well as the relations in which they are engaged. It is, however, conjectured that the proposed approach is feasible even for the organisms with higher cognitive capabilities, i.e. capable of processing more complex scenes. In particular the well known study by Miller in Miller (1956) indicates that for example humans have a bound on the number of simultaneously processed items (chunks). Thus, even very complex scene can be modeled using finite number of parameters. However, the complexity might be all in finding the right chunks to represent the situation at any point in time.

1.3. Structure of the article

The article is structured as follows. Section 2 presents an overview of the related work relevant to the context of this article. The conceptual overview of the proposed learning system is described in Section 3 along with the description and the results of several experiments with honey bees used to showcase the system. Section 4 provides the fundamentals of the theory of Vector Symbolic Architecture relevant to the scope of the article. The details of the design of the artificial learning system follows in Section 5. The results of learning performance evaluation are presented in Section 6. The conclusions and open research issues are presented in Section 7.

2. Related Work

The first clear results providing the evidence that honey bees can learn visual stimuli, which are rules related, were documented in Giurfa et al. (2001). This work showed that honey bees are able to learn the *sameness-difference* concepts for visual stimuli via a delayed matching to sample task where an individual bee had to view a given

target stimulus (e.g yellow disc), load this information into working memory, and subsequently use this information to choose between two alternatives (e.g. yellow disc versus blue disc). Bees had to learn to use the rule of matching test stimuli to sample since the nature of the visual stimuli continued to change during trails, and importantly once bees had acquired this information regarding the rule in the visual domain the rule could be applied to novel stimuli presented in the olfactory domain. The bees could also learn the opposite rule (i.e., do not match the sample), demonstrating a remarkable degree of plasticity for learning problem solving in bees. Later in Avarguès-Weber et al. (2011) it was shown that bees are capable of learning the *above-below* relation in visual stimuli and able to apply learned relation to novel stimuli in transfer tests. Avarguès-Weber et al. (2012) presents the results of experiments showing that honey bees can master acquiring of two concepts simultaneously, namely a *above-below* or *left-right* rule as well as *sameness-difference* rule between the different components of the stimuli. Importantly, individual bees could simultaneously acquire both levels of this rule based problem despite the fact that only a single level might act as a predictor of reward; suggesting bee cognition evolved to learn maximum information content about relevant stimuli. It is likely that such impressive learning capabilities result from allocation of attention to various components of stimuli, as bees receiving different levels of priming to either local or global information content within a complex scene modulate their choices to the most relevant components Avarguès-Weber et al. (2015). An overview of achievements in studying conceptual learning by miniature brains is presented in Avarguès-Weber et al. (2013). Recent work Avarguès-Weber et al. (2014) shows an ability of honey bees to discriminate and learn relations based on relative sizes of objects in the visual stimuli, and having learned the rule bees were able to apply it to novel stimuli varying is shape, size and color. Potential usage of these findings for computer vision is discussed in Dyer et al. (2014), highlighting how insights from insect evolved systems can provide important clues for software design of what sensory channels are required to solve complex problems. An important conclusion is that the rule learning could be a useful ability for computer vision systems, because learned rules can be applied to novel previously unseen scenarios. Similar discussion about concept learning in neuromorphic vision systems can be found in Sandin et al. (2014). It discusses potential advantages of combining different techniques such as neuromorphic systems, achievements in studying capabilities of miniature brains of insects, distributed computer vision systems, wireless data transmission and Vector Symbolic Architecture. The paper sets an important framework for how it might be possible to model the cognitive behaviors of real brains as found in honey bees, but the challenge has been to implement such complex biological systems that can learn through experience.

Distributed data representation is widely used for computer based semantic reasoning Kanerva (2009), Plate (2003). Examples of cognitive capabilities of distributed representations include solving Raven's progressive matrices Rasmussen and Eliasmith (2011), Emruli et al. (2013) and Levy et al. (2014). Vector Symbolic Architecture (VSA) is a bio-inspired representation of structured knowledge. VSAs are inspired by brain activity where simple mental events involve the simultaneous activities of very large number of dispersed neurons Kanerva (2009). Information in VSA is represented distributively, where a single concept is associated with a pattern of activation of many

neurons. In VSA this is achieved by using codewords of very large dimension. There are several different types of VSA using different representations, e.g. Plate (2003), Kanerva (2009), Gayler (1998), Rachkovskij and Kussul (2001), Aerts et al. (2009), Rachkovskij et al. (2013), Gallant and Okaywe (2013), Snaider and Franklin (2014). This article utilizes a subclass of VSA based on so-called Binary Spatter Codes Kanerva (1997). It was recently shown in Levy et al. (2014) that VSA can address three central problems posed by Wittgenstein: rule-following, aspect-seeing, and the development of a "private" language. Moreover, there are several works where VSAs are applied to certain application areas. In Gallant and Okaywe (2013) a VSA-based knowledge-representation architecture called MBAT is proposed for exploitation with machine learning algorithms. Recent applications of VSA for analysis of generic patterns and symbol sequences are proposed in Kleyko et al. (2015a), Kleyko and Osipov (2014a), Kleyko and Osipov (2014b), Kleyko et al. (2015b). In Räsänen and Kakouros (2014) authors present VSA-based approach for modeling dependencies in multiple parallel data streams. A description of architecture, which utilizes VSA to achieve interoperability of systems with nearly zero configuration, can be found in Emruli et al. (2014).

In Levy et al. (2013) a VSA-based knowledge-representation architecture is proposed for arbitrarily complex, hierarchical, symbolic relations (patterns) between sensors and actuators in robotics. However, in contrast to the proposed approach the article showcases the representation of preprogrammed relations between sensors and actuators rather than learning from the observed environment.

3. Outline of the proposed online concept learning architecture

This work proposes a simple, yet extensible, online learning architecture functionally imitating the recently observed concept learning process in honey bees Avarguès-Weber et al. (2012). The proposed VSA-based pipeline is illustrated in Figure 1. The novel part and the contribution of this article is the functionality of the *stimuli encoding block*, which is used to represent the observed episode as a set of relations between different objects. The *stimuli encoding block* is graphically outlined in Figure 2.

The stimuli encoding block takes advantage of two useful properties of VSA. Firstly, VSAs can represent relationships between arbitrary, novel entities without requiring training. The representational capability arises directly from the structure of the VSA network - so does not require the extensive training of techniques like backpropagation. Secondly, VSAs allow the representation of multiple relationships simultaneously on the same set of neurons, once again without specific training to enable it. These two properties of VSA enable the neural network to deal novel combinations of relationships without requiring further training.

The core of the *stimuli encoding block* is the *pool of relations' representations or templates* described using VSA. While the details of VSA are presented in the next section for the purposes of this section it is sufficient to say that VSA is a class of connectionist models for representation of structured knowledge using, e.g. binary codewords of very high dimensionality, i.e. thousands of bits (throughout the articles terms *HD-codes*, *HD-vectors* or *HD-codewords* are used when referring to a codewords of the VSA class Kanerva (1997)). The knowledge structuring is expressed using basic

arithmetic operations on the codewords, which encode the concepts. The reasoning and inferences are due to unique statistical properties of long codewords Kanerva (2009), Plate (2003).

It is assumed that the system operates on *episodes* of visual inputs, where an episode is all visual input appeared during one learning cycle, which begins with presenting the system a visual training sample and ends by presenting the system a positive or a negative reward. For each episode vision and feature extraction circuitries are able to isolate entities in the environment, detect binary relations between those entities, and return vectors encoding the identities of the entities, the identity of the relation, and the roles that the entities play in that relation. Obviously, the system relies heavily on the capabilities of the visual circuitry, but this will be true of all visual organisms. The question of interest here is not how much is done by the specialized visual circuitry, but how to combine the outputs of the visual circuitry to generated rule-following behavior. VSA provides a good mechanism to do this.

All representations generated by the pipeline are used for encoding particular instances of relation representation of both spatial and comparative types are stored in the *item memory*. These are then encoded into two different forms of VSA representation: the *bonding* representation, which captures the relational relation between the entities, and the *comparative* representation, which captures the relation between the corresponding features of the entities. The bonding representation is notated as:

$$bond_i(re_i, e_{i,1}, e_{i,2}) = [re_i + ro_1(re_i) \oplus e_{i,1} + ro_2(re_i) \oplus e_{i,2}] \tag{1}$$

Note that in all equations the following notation for arithmetic operations are used. $\oplus$ denotes the bitwise XOR operation[1]. Notation $[A + B + C]$ is used for "majority summation" (also explained in the next section), which implements operation of vectors' bundling in VSA.

The naming of variables and functions used in (1) and (2) is mnemonic in order to ease the understanding. Thus in (1) all the arguments of $bond$ are HD-vectors supplied by the visual and feature extraction circuitry. The subscript i indicates the episode being processed by the visual circuitry. The HD-vector re indicates the identity of the binary relation detected by the visual circuitry. This is one of a small number of discrete values (codewords) indicating each of the relations that can be detected by the visual circuitry. Similarly, the ro functions (ro_1 and ro_2) return values that are the codewords representing each of the two roles of the binary relation represented by re. Conceptually, the HD-vectors re, $ro_1(re)$, $ro_2(re)$ should be generated by the visual circuitry, but it makes the intent of the VSA representation easier to understand by treating the generation as a part of the representation.)

The bonding representation can for example be used to encode a *spatial* relation between a pair of objects. For instance, for the the vertical alignment spatial relation labeled "above-below" the roles might be labeled "upper-entity" and "lower-entity". Note that these labels are purely for expository convenience. Similarly for the horizon-

[1]While in the VSA- related literature $\otimes$ sign is used to denote operation of vector binding, here in order to avoid possible confusion with tensor multiplication operation, sign $\oplus$ is used to reflect the actual arithmetical operation implementing binding.

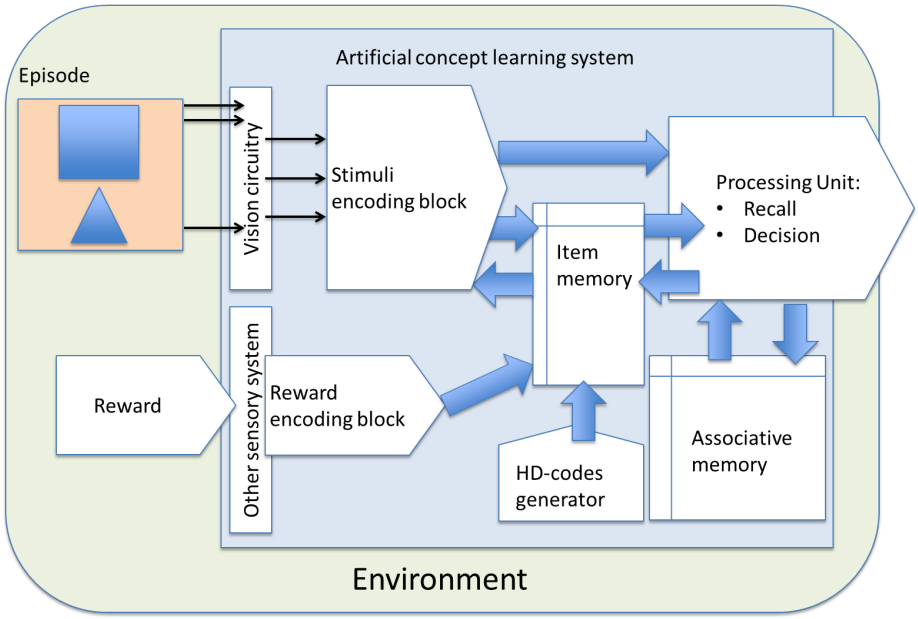

Figure 1: A simple concept learning architecture illustrating interactions of the stimuli encoding block with item memory, associative memory, recall, and processing units.

tally alignment relation labeled "left-right" the roles might be labeled correspondingly "right-entity" and "left-entity". The actual values are arbitrary HD-vectors.

The comparative representation is notated as:

$$comp_i(f, e_{i,1}, e_{i,2}) = [comprel(f) \oplus comparison_f(feat_f(e_{i,1}), feat_f(e_{i,2}))] \quad (2)$$

In (2) all the arguments of $comp()$ are HD-vectors supplied by the visual and feature extraction circuitry. The subscript i indicates the episode being processed by the visual circuitries. The function $comprel(f)$ returns the HD-vector that indicates the identity of the binary relation that is appropriate for comparison of values of feature f. The $feat_f$ functions return the value of the feature f for the argument entity e. For example, for $f = shape$ the returned values might be *circle* or *square*. The $comparison_f$ function returns the HD-vector representing the result of comparing two values of feature f. For example, for $f = shape$ the comparison of all possible shape value pairs might yield *same* or *different*. Again, these labels are purely for expository convenience. The actual values are arbitrary HD-vectors.

Representations (1) and (2) should be viewed as templates in a sense that at the time of the system initialization no particular meaning is given to the specific representation, also the objects to be encountered in the relations are not known. In other words the system starts with the clean slate state. Further in the article the terms representation and template are used interchangeably.

In the case of honey bees, which are used as the model for this study, whilst the entire range of their potential rule learning capacities is yet to be tested, the experiments

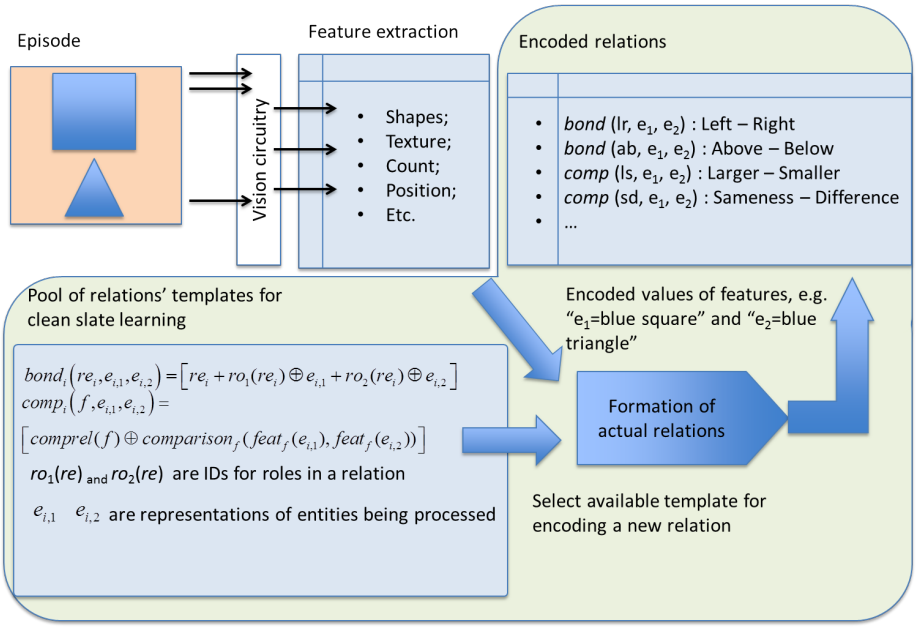

Figure 2: Details of the stimuli encoding block. A square shape above and a triangle shape below form the visual episode for the input

in Avarguès-Weber et al. (2011, 2012, 2014) proved that they are capable of processing at least the following relations: *sameness-difference*; *above-below*; *left-right*; *larger-smaller*. Note that *sameness-difference* is defined with respect to specific properties of the images and is, therefore, the relation across any pair of attributes. The outlined above pipeline is not, however, limited in the number of potentially recognizable relations. The number of the recognizable relations is determined by the complexity of the sensory and the feature extraction system on the one hand and the size of the memory and computational power of the processing unit on the other. This observation rather logically follows from the diversity of the intelligence level of the known living organisms.

When an episode is processed, that is all entities and their relations are processed and encoded into bonding and comparative representations, a complete representation of the episode is constructed by *bundling* all encoded representations as:

$$\textbf{EPISODE}_i = [bond_i(\ldots) + comp_i(\ldots)]. \tag{3}$$

For example, assume an episode consists of two vertically aligned triangles (T1 and T2) of the same color but of the different size ($sizeof(T1) < sizeof(T2)$). The bonding spatial representation for this episode is then encoded (with mnemonic labels) as $bond(above-below, T1, T2) = [above-below + above \oplus T1 + below \oplus T2]$; the comparative representation is encoded as $comp(size, T1, T2) = size \oplus different$; finally, the whole episode is represented as $\textbf{EPISODE} = [bond(above-below, T1, T2) + comp(size, T1, T2)]$.

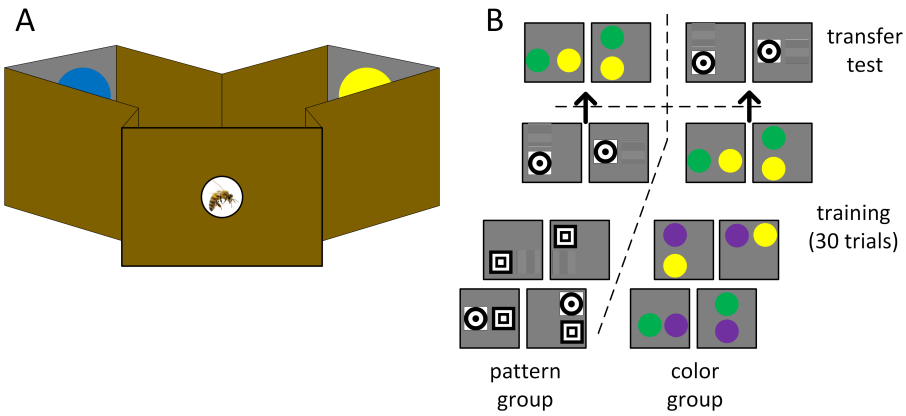

Figure 3: **A**. The maze used in experiments with honey bees. **B**. Stimuli used in experiments with honey bees. Two types of stimuli were used: pattern group and colour group. Honey bees have to choose between two visual stimuli. During the training trails either *above-below* or *left-right* relations are corresponding to the reward. Opposite relation leads to quinine solution.

The episode's representation (3) is then used by the *processing unit* for *experience* formation during the training phase of the system as well as for the recall operation and decision making in the operating phase. To complete the process and represent the system's *experience* the processing unit awaits the arrival of the *reward* signal. Upon the arrival of the reward signal the *processing unit* binds it with the representation of the episode (e.g. **EPISODE** $\oplus$ $textbfREWARD$) and updates the *associative memory*. During the system's lifetime this experience is used for online reasoning and decision making processes. The detailed procedure is described in Section 5.

3.1. The showcase: Description of the experiments with honey bees

The proposed architecture for artificial concept learning is inspired by the experiments with free flying honeybees as described in Avarguès-Weber et al. (2012). The results of these experiments are also chosen for the benchmarking the performance of the proposed solution. This subsection summarizes the major aspects of the honey bee training experiments. Its setup is presented in Figure 3. The bees were trained to fly into a Y-shaped maze, which presented two alternative paths. A bee has to choose either the left or the right one. Paths are indicated by different visual stimuli. One of paths leads to reward (sucrose), while another one leads to penalty (quinine). Obviously, honey bees are motivated to collect as much reward as possible in this context, and avoid the quinine Avarguès-Weber et al. (2010). Thus the learning objective for a bee is to correctly infer the path leading to sucrose solution every time when the maze is entered.

Examples of visual stimuli presented to bees are illustrated in Figure 3. In Avarguès-Weber et al. (2012) a visual stimulus comprises two concepts that can be learned. One represents spatial relation between shapes (*above-below* or *left-right*). The second learned concept *sameness-difference* between features of the shapes in the episode.

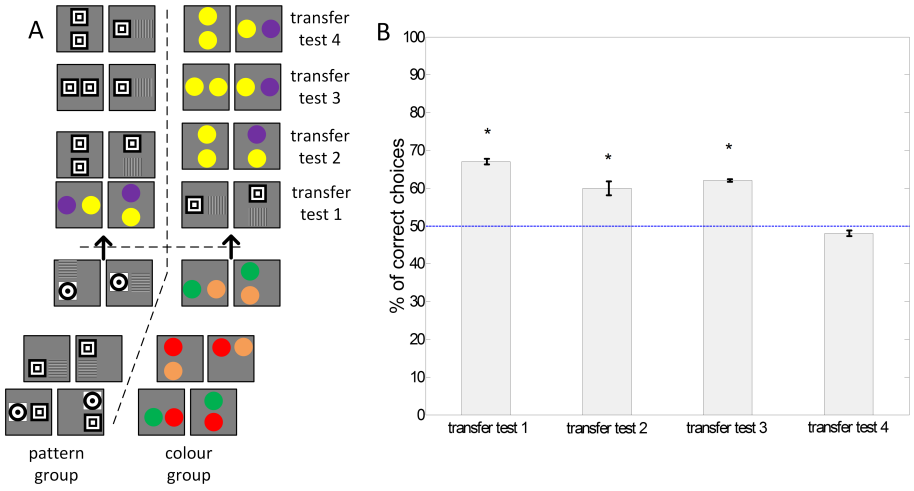

Figure 4: **A**. Bees were trained in a maze to choose between stimuli presenting two different patterns (group 1) or two different colored discs (group 2) in an above-below (or right-left) relation depending on the group of bees. Transfer tests are shown for the above-below relation. **B**. Four transfer tests assessing learned concepts. Every test had 30 trials. In transfer test 1, *correct choices* indicate choice of the previously rewarded spatial relation; in transfer tests 2 and 3 and 4, the term indicates choice of the stimulus with 2 different images. In transfer test 1, bees transferred their choice to unknown stimuli presenting the appropriate spatial relation despite belonging to a different modality. Transfer tests 2–4 demonstrated that bees also learned that the stimuli had to present two different images. Symbol * indicates that the percentages of the correct choices are significantly different from 50%. Adopted from Avarguès-Weber et al. (2013).

The difference is indicated either by different colors or by different hatches. The honey bees were trained during 30 trials on each type of the stimuli, e.g. achromatic shapes. Possible bias in the selection bees' choice was avoided by varying sides of the rewarded and non-rewarded stimuli in each trial. For the same reason appearances and relative positions of the shapes varied from trial to trial. An interested reader is referred to Avarguès-Weber et al. (2012) for more details.

In order to assess the quality of the concept learning by honey bees four different transfer tests were performed. The layout of the tests and corresponding results are presented in Figure 4. During the experiments one half of bees was trained for *above-below* relation and another one for *left-right* relation. During training trials the *sameness-difference* concept was always represented by two different colors or two different patterns. The first transfer test shows that bees are able to successfully apply previously rewarded spatial relations to unseen stimuli. For example for the group trained with achromatic figures colored circles were presented (Figure 4). Second and third transfer tests show, that bees, presented with stimuli comprising the same spatial relation (either right or wrong), prefer stimuli with learned sameness/difference concept. The last test revealed that both concepts are equally important in the learned experience. In other words bee's choice was random when one stimuli had proper spacial concept while another had proper difference concept. Therefore, bees used both rules simultaneously.

4. Fundamentals of Vector Symbolic Architecture

As mentioned previously, VSA is an approach for encoding and operations on distributed representation of knowledge, and has previously been used mainly in the area of cognitive computing and natural language processing, see Kanerva (2009), Plate (2003) and Gallant and Okaywe (2013) for details. The fundamental difference between the distributed and traditional (localist) representations of data is as follows: in localist computing architectures each bit and its position within a structure of bits are significant (for example, a field in a database has a predefined offset amongst other fields and a symbolic value has unique representation in ASCII codes); whereas in a distributed representation all entities (and structured combinations of entities) are represented by vectors of very high dimension. That is, entities are represented by the direction of a vector in a high-dimensional space and every bit contributes to defining the direction of the vector. In a localist representation single bits or small groups of bits can be interpreted without reference to the remainder of the bits. In a distributed representation it is only the total set of bits that can be interpreted. For the remainder of the article the term HD-code is used when referring to such codes. *High dimensionality* refers to that fact that in HD-codes, several thousand positions (of binary numbers) are used for representing a single entity; Kanerva (2009) proposes the use of vectors of 10000 binary elements. Such entities have the following useful properties.

4.1. Randomness

Randomness means that the values on each position of an HD-code are independent of each other, and "0" and "1" components are equally probable. In very high dimensions, the distances from any arbitrary chosen HD-code to more than 99.99 % of all other vectors in the representation space are concentrated around 0.5 normalized Hamming distance. Interested readers are referred to Kanerva (2009) and Kanerva (1988) for comprehensive analysis of probabilistic properties of the hyperdimensional representation space.

In VSA, information is carried by the relations (angles) between vectors rather than the absolute directions of the vectors. Thus, the choice of vectors representing 'atomic' entities is arbitrary. If the atomic entities should be unrelated (i.e. because there are no initially learned relations in order to avoid bias it should be assumed that the entities are unrelated) the HD-codes representing them should be chosen so that they are mutually orthogonal (maximally dissimilar). In high-dimensional spaces two vectors chosen at random will be close to orthogonal with very high probability. That is, choosing at random the vectors to represent atomic entities yields the desired properties.

4.2. Similarity metric

A similarity between two binary representation is characterized by normalized Hamming distance, which (for two vectors) measures the number of positions in which they differ divided by the size of the vectors:

$$\Delta_H(A, B) = \frac{1}{d}\|A \oplus B\|_1 = \frac{1}{d}\sum_{i=0}^{d-1} a_i \oplus b_i, \tag{4}$$

where a_i, b_i are bits on positions i in vectors A and B of dimension d, and where $\oplus$ denotes the bit-wise XOR operation.

4.3. Bundling of Vectors

Joining several entities into one structure is done by the bundling operation; it is implemented by a thresholded sum of the HD-codes representing the entities. A bit-wise thresholded sum of n vectors results in 0 when $n/2$ or more arguments are 0, and 1 otherwise. In the case of an even number in sum ties are broken at random. This is equivalent to adding an extra random HD-code Kanerva (2009).

Furthermore terms "thresholded sum" and "majority sum" are used interchangeably and denoted as $[A + B + C]$. The relevant properties of the majority sum are:

- the result is a random vector, i.e. the number of '1' components is approximately equal to the number of '0' components;
- the result is similar to all vectors included in the sum;
- as the number of HD-codes that are operands of the majority sum operator increases, the Hamming distance between the result of the operation and any of the operands tends to 0.5 (which is the expected similarity of any two HD-codes selected at random). That is, the information capacity of the result vector is finite and information about the operands is lost as more of them are combined.

4.4. Binding of vectors

Binding, which can be interpreted as assigning a value to a field, is implemented by bit-wise XOR operation on corresponding vectors. The important properties of XOR operation are:

- the binding is invertible (unbinding), i.e. if $C = A \oplus B$ then $C \oplus A = B$;
- the binding distributes over bundling
  $D \oplus [A + B + C] = [D \oplus A + D \oplus B + D \oplus C]$;
- when the operands have a density of 0.5 (equal number of 1 and 0 bits) the result has an expected density of 0.5.
- the result is dissimilar to both operand vectors: $\frac{\|B \oplus (A \oplus B)\|_1}{d} = 0.5$ and $\frac{\|A \oplus (A \oplus B)\|_1}{d} = 0.5$. However, binding preserves relational similarity: $\Delta_H(A, B) = \Delta_H(C \oplus A, C \oplus B)$.

4.5. Composition and decomposition of structures in distributed representation

Representing a compositional structure containing roles (fields) and fillers (values) to describe relations such as (1), requires both VSA operations: binding for association of roles with fillers and bundling for joining the set of bound role-filler components. Thus, a compositional structure S with three roles (fields) and associated fillers (values) is defined as: $S = [R_1 \oplus F_1 + R_2 \oplus F_2 + R_3 \oplus F_3]$.

The extraction of information out of distributively represented structures is the reverse of the encoding process. Namely, a particular vector-filler is extracted by XORing

the HD-code representing the structure with the desired vector-role. This follows from the self-inverse property of binding with XOR.

Consider the following example for better clarity. To decode filler F_1 from compositional structure S the following set of operations is performed on S:

1. XOR with HD-code representing role R_1 in order to retrieve its filler: $F_1' = S' \oplus R_1 = F_1 + NOISE$.
2. Pass noisy vector F_1' to the item memory for finding the best clean match, i.e. search for an entry with minimal to F_1' normalized Hamming distance.

Note that in the VSA's context $NOISE$ is not the same as a random signal. In VSA everything (both atomic and composite) is represented by a vector. The direction of the vector represents the identity of the thing represented and the magnitude of the vector represents the strength of the evidence for (or degree of belief in) the thing represented by the vector. The natural basis vectors (the dimensions) of the vector space have no intrinsic meaning. Therefore, there is nothing intrinsic to a vector to indicate whether it represents an atomic or complex object. The only way a VSA system can recognize a vector as representing a composite entity is by being able to decompose it into its components. This requires that the components have a special functional status in the system. This is done by providing a clean-up or item memory. By storing some vectors in this memory they are given a distinguished status, which means that the system will try to recognize arbitrary vectors as composites of the items in clean-up memory. In performing an unbinding like $A \oplus [A \oplus B + C \oplus D] = [(A \oplus A) \oplus B + A \oplus C \oplus D] = [B + A \oplus C \oplus D]$ the operation will generate terms like $A \oplus C \oplus D$ which are legitimate vectors but do not correspond to any entity system would ever want to represent (whereas B does). Therefore, terms like $A \oplus C \oplus D$ are noise in the sense that they do not correspond to a representation system would ever be interested in. They are filtered out by using a clean-up or item memory populated with vectors like *A*, *B*, *C*, and *D*. There is a range of possibilities for clean-up memories. At one end it can be interpreted as a projection onto the subspace spanned by the items. This allows a vector to be cleaned-up to a weighted sum of the items in the memory. If the memory contained *A*, *B*, *C*, *D* the vector $[pA + qB + X + A \oplus B]$ would be cleaned up to $[pA + qB]$ (where p and q are scalar multipliers). At the other extreme there can be competition (equivalent to lateral inhibition) between the retrieved vectors so that only the item with the highest magnitude survives. In the example above the result would be $[A]$ if $p > q$.

4.6. Architectural implementation of VSA operations

The algebra of VSA includes other operations, such as permutation of vectors. Since these operations are not needed in the scope of this article the description of their properties is omitted. The most important point to realize is that all these VSA operations (e.g. binding, bundling, composition, decomposition) are implemented architecturally. That is, they are a mathematical and systematic consequence of simple arithmetic operations on vector spaces, which can be easily implemented by the patterns of connection of the neurons.

Given the correct pattern of neural connections these operations will be applied uniformly and systematically to all representational vectors, even novel ones. Contrast

this with the behavior of traditional backpropagation neural networks, which lack this property of systematicity. An existing operation will not be generalized to novel inputs without extensive training.

4.7. Learning by example in VSA

Capability of HD-codes to learn mappings between concepts from example was firstly presented in Kanerva (2000) and similar ideas were later exploited in Emruli et al. (2013) and Emruli and Sandin (2014). This follows from the self-inverse binding property of Binary Spatter Codes (and some other VSAs). $A \oplus B$ represents the binding of *A* with *B*, but it can equally be interpreted as a representation of a mapping from *A* to *B* and vice versa. Binding $A \oplus B$ with *A* yields *B*, hence, $A \oplus B$ can be interpreted as the representation of a mapping that when applied to *A* transforms it to *B*. The mapping is bidirectional, so it can also transform *B* to *A*. This bidirectionality can be avoided if necessary, but is outside the scope of this paper.) Learning by example can be seen as relational learning. In the scope of this paper learning by example is a core functionality of concept learning block. It is based on the property of VSA to keep structural similarity of knowledge structures. In essence the key component is a vector **M**, which acts as a mapping structure to infer similarity between compositional structures. It can be demonstrated on example from Kanerva (2000). It is known, that if a is the mother of b, then a is the parent of b. Both data structures are represented using the approach from subsection 4.5. For mother relation $\mathbf{mother}_{AB} = [\mathbf{mother} + \mathbf{mot} \oplus \mathbf{A} + \mathbf{child} \oplus \mathbf{B}]$, where **mother** is HD-code representing relation for mother, **mot** and **child** are HD-codes for roles in this structure. For parent relation $\mathbf{parent}_{AB} = [\mathbf{parent} + \mathbf{par} \oplus \mathbf{A} + \mathbf{child} \oplus \mathbf{B}]$, where **parent** is HD-code representing relation for parent, **par** and **child** are HD-codes for roles in this structure. Vector M in turn is constructed as binding of $\mathbf{parent}_{AB}$ and $\mathbf{mother}_{AB}$, i.e. $\mathbf{M} = \mathbf{M}_{AB} = \mathbf{mother}_{AB} \oplus \mathbf{parent}_{AB}$. Thus the mapping vector is based on example. For more examples mapping vector is updated in a similar way: $\mathbf{M} = [\mathbf{M}_{AB} + \mathbf{M}_{CD} + \mathbf{M}_{EF}...]$, where *c*, *d*, *e*, *f*, are new examples of the same relations. Thus, it is possible to generalize that x is the parent of y given only that "x is the mother of y" in form $\mathbf{mother}_{XY}$ by unbinding $\mathbf{M} \oplus \mathbf{mother}_{XY}$ for previously unseen fillers x and y, because the result of unbinding will be most similar to $\mathbf{parent}_{XY}$. This behavior is due to the fact that the same literal A has been in both relations, $mot \oplus A$ and $par \oplus A$. The binding M_{AB} contains terms like $(mot \oplus A) \oplus (par \oplus A) = mot \oplus par$, which implement the mapping of the roles independent of the new fillers.

The power brought to this task must be stressed. Unlike learning in classical neural networks this does not require thousands of training episodes to slowly adjust synaptic weights. Rather, the VSA mapping vector can be learned from a few examples, each seen only once. Each example constructs a representation in a single feed-forward pass through the network and the few examples are accumulated in a short-term memory system. Furthermore, the VSA mapping vector can be successfully applied to novel relational examples. The only requirement is that the atomic roles and fillers are already stored in the clean-up memory.

In the scope if this paper learning by example is used by the processing unit to learn concepts. Mapping structures for different concepts stores bindings of the representations of reward/punishment and HD-codes representing episodes. New precedents

identified for a particular concept are added into mapping vector for this concept. The example of concept formation and processing is presented in the next section.

4.8. Anchoring the proposed VSA-based online concept learning architecture to psychological models of similarity

The analysis of the similarity is a mature psychological discipline with a set of established models. The goal of this section *is not* to formally proof the strict correspondence of the VSA-based concept learning pipeline to the general form of analogical similarity. Rather the intention is to demonstrate that the proposed VSA-based concept learning approach bear some notorious properties of the analogical similarity models, which make them suitable for functional modeling of the concept learning by bees.

The taxonomy of the similarity models include three main classes: Spatial models, featural models and structural mapping. In Markman and Gentner (2005) it is argued that all three models could be used in combination for holistic modeling of cognition. In what follows the main postulates of the models are introduced. According to *spatial models* Shepard (1962); Nosofsky (1986) concepts are represented as points or vectors in a semantic space. Similarity involves finding the distance between points using some scalar metric, which is used to model similarity judgments. While main advantage of this model is low computational complexity, it is, however, impossible to access the specific commonalities and differences that form the basis of the similarity judgment.

Contrast model, also called *featural approach* Tversky (1986). overcomes the shortcomings of the spatial models by representing entities by sets of features. To decide about the similarity pairs of features are compared using elementary set operations. The intersection of sets of features contains commonalities, while features not in the intersection of sets are differences. Calculation of the similarity in the contrast models is computationally inexpensive: Each feature in the representation of one item simply needs to be matched against the set of features representing the other item.

A third approach to similarity called *structural mapping* Gentner (1983) accounts for the importance of relations in similarity and shows the best match to human phenomena. In this class of models the representation consists of entities, attributes and relations. Relations are representational elements that relate two or more arguments, e.g. a relation $above(x, y)$ represents the *above-below* relation between objects x and y. Comparing pairs of relational representations requires their satisfaction to the structural constraints. The structural consistency comprises the constraints of *parallel connectivity* and *one-to-one mapping*. Parallel connectivity means that if two relations are compared, then their arguments must match. For example, if two $above(x, y)$ relations are compared then things on top and things on the bottom must be matched correspondingly. One-to-one mapping means that each element in one representation matches to at most one element in the other. For example when two relations, say *above-below* are compared and both contain the same objects in conflicting states (e.g. a square in the first relation is located above and the same square in the other relation is located below) these two objects should be treated as different in the *above-below* test. The structure mapping model is much more computationally intensive than spatial or contrast models. The calculation of parallel connectivity requires checking each correspondence among relations to ensure that the arguments of those relations also match. Also to

enforce one-to-one mapping constraint may require a number of comparisons among potential matches.

While the three models overviewed above are different in their expressive power, Markman and Gentner (2005) argues that their combination could capture the different aspect of cognitive processing. Different modes of the similarity processing include: Fast judgment of the degree of similarity (spatial and contrast models); and Slower, more computationally expensive but richer comparison, giving an access to the commonalities and differences by the structural mapping. There are psychological evidences that the similarity analysis process in humans use both assessments modes. Computationally simpler forms of similarity operate all the time searching for similarities among items in the environment as well as between items in the environment and background knowledge. The results of the simple comparisons are used to guide the allocation of resources for more detailed similarity analysis.

4.9. Reflection of the structural mapping similarity model to the case of VSA-based online learning architecture

Leaving the in-depth analysis of the correspondence of the proposed VSA-based concept learning approach to the general models of analogical similarity outside the scope for this article this section resorts to drawing notable parallels between their main postulates and the design choices made in this article.

Prior to all one should stress the *dualism* of distributed representations and Vector Symbolic Architectures. On the one hand VSA is an approach to encode structured information. See section 4 for details. On the other hand the usage of the distributed encoding approach (e.g. high-dimensional binary codes and representing structures by basic binary arithmetic operations) enables comparison of structured information using a scalar metric (Hamming distance). This dualism enables the first parallel between the proposed approach and the psychological theories of similarity. Namely, while VSA could to a certain extend be used for structural mapping analysis it also bears some functional properties of spatial similarity models.

The comparative representation of relations (2) is used to encode the presence feature-based similarities or differences in the represented episode as in the contrast similarity models. Their similarities and differences among features are detected by the features' extraction block by elementary comparison operations. The comparative relations could be either used stand-alone or in combination with the bonding representations (1) for the purposes of the quick assessment of the episode. The latter usage is adopted in the proposed online concept learning system.

Note that since both spatial and featural similarity models have implicitly assumed low-dimensional spaces they are not strictly mappable to the case of high-dimensional spaces. Here a functional parallel is drawn as the VSA-encoded relations are essentially point in high-dimensional space of binary distributed representations their similarity could be quickly assessed by by computing a scalar distance metric. If the vectors representing the stimuli are scattered over a non-trivial subspace of the total vector space their similarity will only be discriminable to the resolution of *same-different*, whereas if the vectors are constrained to fall in a sufficiently low-dimensional subspace their similarity will be calculable as a graded quantity.

The notion of relations is central to the proposed system as in the structural mapping theory. As the result of the episode's processing the feature extraction block determines the entities, their attributes and relational properties. Note that structural similarity relies on subgraph isomorphism between the two representations to determine the correspondences between entities and relations. A weaker form of the structural similarity would use literal similarity of relational roles to determine the correspondence between fillers. The proposed system suggests two kinds of relations' templates: the bonding and the comparative templates.

The bonding representations are used to represent proper directional relations. That is the relation $bond(above - below, x, y) = [above - below + above \oplus x + below \oplus y]$ encodes relation "entity x is above entity y". This form of the relation's representation ensures both the parallel connectivity and the one-to-one mapping constraints. These constraints are ensured by virtue of how the representations are used, i.e. by binding them together. Given the self-inverse-binding property of high-dimensional binary spatter codes the repeated literal values serve to "align" the corresponding elements. In fully general analogical mapping, however, both the fillers and the roles may differ between the structures being compared, therefore this alignment mechanism would not be sufficient.

The parallel connectivity is ensured by the comparative inclusion of the directionality in the VSA representation of the relation. That is for the (detailed) comparison of two relations of the same type, one first needs to unbind *in both relations* the HD-codes for the objects which are located above (by performing XOR operation with HD-code for $above$), then to unbind the HD codes for the objects which are located below (by performing XOR operation with HD-code for $below$) and finally compare the results pairwise, i.e. above with above and below with below[2].

To demonstrate the functional satisfaction to the one-to-one mapping constraint suppose the system encountered another relation of the same type but where object x is located below another object, say z, i.e. $bond(above - below, z, x) = [above - below + above \oplus z + below \oplus x]$. Obviously, despite the fact that HD-codes for object x in both cases are the same, the result of the relations' encoding would be different since binding x with a codeword for $below$ would produce a dissimilar result than that when binding x with a codeword for $above$ due to mutual dissimilarity of $above$ and $below$. Note again that this discussion should not be interpreted as the direct correspondence to the one-to-one mapping in the analogical mapping literature. There this constraint means mapping between fillers, i.e the constraint is that if filler x maps to filler y in one part of the mapping it should do so everywhere in the mapping. In the case of the proposed pipeline the role-filler binding is conceived as a mapping, but that is strictly speaking a different type of mapping from the filler-filler mappings of the analogical mapping.

To summarize, the form of VSA relations' representations (1) and (2) as well as the representation of the episode (3) is chosen because of:

[2]Note that this is a very serial computation way of using the representations. It was chosen mainly for demonstration of functional satisfaction of the parallel connectivity constraint by the proposed approach. One of the big advantages of VSA is that it allows holistic computation.

Table 1: Notations used in Section 5.

Symbol	Definition
ab	re for above-below relation
a	ro_1 for above role
b	ro_2 for below role
ls	Result of *comprel(f)* for larger-smaller relation
sd	Result of *comprel(f)* for sameness-difference relation
equal_size	HD-code representing equality in the larger-smaller feature-relation (result of *comparison*)
different	HD-code representing difference of objects in the same-different feature-relation (result of *comparison*)
EXPERIENCE	mapping vector for concept learning
REWARD	HD-code for reward (sucrose)
Cy	Encoded values of features for yellow circle
Cp	Encoded values of features for purple circle
STIMULUS$_i$	Representation of ith stimulus

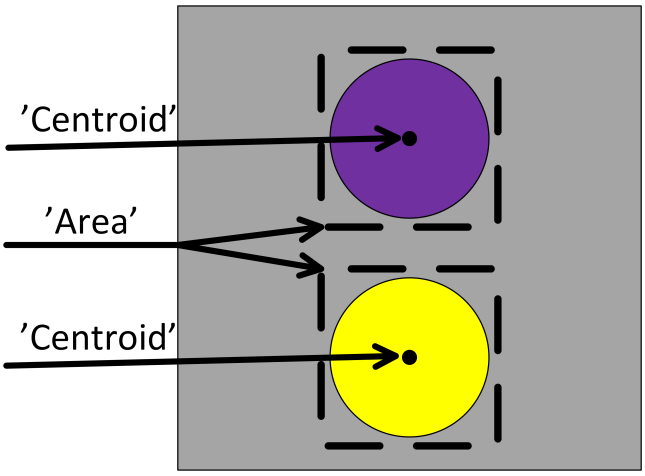

Figure 5: An example of a visual stimulus and the output of the *vision circuitry* implemented by *regionprops()* function of MATLAB.

1. Experimental evidences that target biological system (honey bees) are capable of both feature-based and relations-based similarity analysis;
2. For quick holistic similarity assessment based on the well-defined scalar metric of similarity for the target decision making scenario (select one of several alternative paths).

5. Design details of the online concept learning system

This section describes the design details of the online VSA-based concept learning system and exemplifies its current implementation replicating the scenario for the honey bees experiments as in Section 3.1. Table 1 introduces the notations used throughout the section.

5.1. Vision circuitry

The first block of the system is a *vision circuitry*. It perceives the episode and extracts the features of the presented visual stimulus. For the purposes of further discussion the set of considered stimuli was restricted to the one used in honey bees' experiments, i.e. geometrical figures (circles) with single-colour fill, see Figure 5. Without claiming any biological plausibility the *feature extraction* block is implemented using the standard image processing toolbox, in the current implementation the feature extraction was implemented by *regionprops()* function in MATLAB. It takes binary image as a logical array, measures and outputs a set of properties for each object in the episode. The relevant for the scope of this discussion properties include 'Area' and 'Centroid'. These parameters are then used to extract higher level features such as: shapes, colours, raw relations (relative spatial placement, relative sizes, etc.). These outputs are then fed to the *stimuli encoding block*.

5.2. Stimuli encoding block

While in general the system does not operate with human understandable labels for relations for simplicity of representation this subsection operates with human readable labels. For example when two 'Centroids' of two objects are different by the horizontal axis these two objects are labeled as being in the *left-right* relation. Similarly when 'Centroids' of two objects are different by the vertical axis they are labeled as being in the *above-below* relation. Depending on the quality of colors (shapes) of the two objects in the points of their 'Centroids' they are labeled as being in the *sameness-difference* relation. Depending on the relative 'Area' properties of two objects they are labeled as being in the *larger-smaller* relation. Note, that a very specific symbolic processing to the "perceptual" output of *regionprops()* has been applied. Given the experimental design of the visual stimuli, one could completely accurately predict the encoding of the stimuli from the experimental condition and the encoding is of the minimal features needed to represent stimuli. Thus, the *regionprops()* visual part is not a part of contribution of the paper. The paper is not investigating any visual stimuli that might interfere with the encoding in any way.

Next, the *stimuli encoding block* generates random HD-codes to encode the objects of the episode and all the features. These vectors are stored in the *item memory*. As the system continues to operate new HD-codes are added to the item memory for each newly detected feature or an object. The representations for repeated items are, therefore, taken from the item memory.

Next, for each new feature-relation and the objects engaged in this relation one VSA template is drawn from the pool for encoding the relation as a VSA representation. This step is best explained by a concrete example of encoding the episode depicted in Figure 5. There, the vision circuitry will detect two objects of the same shape (circle), the objects are of equal size, the colours of the objects are different and that the purple circle is located above the yellow circle. Assuming the system's item memory is initially empty, the *stimuli encoding block* will generate HD-codes as summarized in Table 1.

The following VSA-relations are encoded:

1. $REL1 = [\mathbf{ab} + \mathbf{a} \oplus \mathbf{Cp} + \mathbf{b} \oplus \mathbf{Cy}]$. This is the encoded *above-below* relation between objects with IDs **Cp** (purple circle) and **Cy** (yellow circle) using a template of type one;
2. $REL2 = [\mathbf{ls} \oplus \mathbf{equal\_size}]$. This is the encoded *larger-smaller* relation using a template of type two;
3. $REL3 = [\mathbf{sd} \oplus \mathbf{different}]$. This is the encoded *sameness-difference* relation using a template of type two;

When all relations are represented in VSA format the encoding block constructs VSA representation of the entire episode, called **EPISODE** as in (3). In the case of the considered example the distributed representation generated from episode in Figure 5 is $\mathbf{EPISODE}_1 = [\mathbf{ab}+\mathbf{a}\oplus\mathbf{Cp}+\mathbf{b}\oplus\mathbf{Cy}+\mathbf{ls}\oplus\mathbf{similar}+\mathbf{sd}\oplus\mathbf{different}]$. When the episode's representation is constructed it is temporarily stored by *stimuli encoding block* awaiting the arrival of the reward signal to complete the process. Note that the particular reward could be associated with many episodes. In this case the system should implement storage of all of them in the temporal memory. This aspect is implementation specific and as such the details are not considered in this article. In the considered example of the experiments with honey bees' the two episodes were involved in the reasoning about the reward.

When the reward signal arrives a VSA representation of the present experience is constructed as $\mathbf{EXPERIENCE} = [\mathbf{EXPERIENCE} + \mathbf{EPISODE}_i \oplus \mathbf{REWARD}_{p|n}]$. For example in the case of three stimuli it will look as $\mathbf{EXPERIENCE} = [\mathbf{EPISODE}_1 \oplus \mathbf{REWARD}_{p|n} + \mathbf{EPISODE}_2 \oplus \mathbf{REWARD}_{p|n} + \mathbf{EPISODE}_3 \oplus \mathbf{REWARD}_{p|n}]$. This representation means that HD-variable **EXPERIENCE** accumulates in itself all VSA representations of stimulus $\mathbf{EPISODE}_i$ leading to the reward encoded by HD-code $\mathbf{REWARD}_{p|n}$ where index "$p|n$" indicates either positive or negative reward. Note that **EXPERIENCE** vector plays the same role as mapping vector **M** in subsection 4.7. The distributed representation of the experience is then stored in the associative memory for later recall (i.e. reasoning).

5.3. Concept recall (reasoning)

The system implements reasoning via the recall of **EXPERIENCE** stored in the associative memory. The recall can be done already after the reception of the first reward. In the recall process the processing unit firstly receives representations for two scenes and constructs relational representation for each of the scene as in (3). For consistency reasons the representations of the scenes in the recall phase will also be called *episodes* and notated as $\mathbf{EPISODE}_1^*$ and $\mathbf{EPISODE}_2^*$. The superscript "*" indicates that the episodes are possibly presented to the system for the first time. Next, it retrieves the current **EXPERIENCE** from the associative memory; then it unbinds representations of stimuli from **EXPERIENCE** as:

$\mathbf{REWARD}_1^* = \mathbf{EXPERIENCE} \oplus \mathbf{EPISODE}_1^*$.

$\mathbf{REWARD}_2^* = \mathbf{EXPERIENCE} \oplus \mathbf{EPISODE}_2^*$.

Finally, the processing unit chooses the episode whose normalized Hamming distance is closer to the target **REWARD**. In other words according to the previous experience the chance to get reward will be higher for that episode.

6. Simulation and results

The goal with the simulations is to assess the accuracy of the learned concepts' recall by the proposed system. As the performance evaluation of the *vision circuitry* is outside the scope for this article, for the simulations 120 visual episodes of similar to these used in the honey bees' experiment Avarguès-Weber et al. (2012) (see Figure 4) were generated and preprocessed, i.e. stimuli were encoded exactly as expected by their design.

It should be emphasized, however, that the system is not attempting to model what is actually happening in the bee's brain. The architecture and any parameters of the architecture are not constrained by the knowledge about honey bees. The only used fact is that bees can encode and use several relational descriptions of the episode. Thus, one should not expect the results to give a quantitative match to the bee's behavior for all possible levels of the visual problem that can be solved by honey bees.

For the simulations each object was not represented by a VSA representation incorporating all individual features. Rather the whole object, e.g. a purple colored circle of size s, was encoded by one randomly generated HD-code. The dimensionality of HD-codes in simulations was 10 000 elements[3]. In total for the simulations 25 HD-codes were generated for encoding colored circles and the features-relations in the episode: 12 for representing objects (colored shapes); 12 in total for representing the relations' templates (3 for *above-below*; 3 for *left-right*; 3 for *large-small*; and 3 *same-different*); and 1 for representing the reward signal.

The objective of the simulations was to evaluate the accuracy of the concept recall after the same number of trials as in the real-world experiments, i.e. 30. For this purpose in the training phase all possible partial permutations of 6 HD-codes representing objects of 2 distinct HD-codes were presented, 30 permutations in total. All 2-permutations of the remaining 6 objects were used to generate 30 test cases for evaluating the accuracy of the concept learning. During each round of the training process the system was presented with two episodes comprising of the same objects but engaged into different spatial relations. The environment generated two reward signals: a positive or a negative. The positive reward corresponds to the situation where a bee found sucrose at the end of the tunnel, the negative reward corresponds to finding the quinine. Only positive rewards were stored in the **EXPERIENCE** VSA in the *associative memory*. For the simulations it was decided to associate the positive **REWARD** with the *above-below* relation. The associative memory was empty in the beginning of the simulation. After training the system on 30 training trials the accuracy of the VSA-based concept learning was tested on 30 new previously unseen cases. In the testing phase two VSA-encoded episodes were used to unbound the **REWARD** representation for each episode from the **EXPERIENCE** VSA recalled from the *associative memory*. The results of the unbinding operation (two noisy versions of the **REWARD**) were compared with the clear version of **REWARD** representing the positive reward, which is stored in the *item memory*. The system chooses the episode revealing the smallest normalized Hamming distance to the clean **REWARD** representation. During

[3]This is a typical dimensionality for VSA simulations (e.g. Kanerva (2009)).

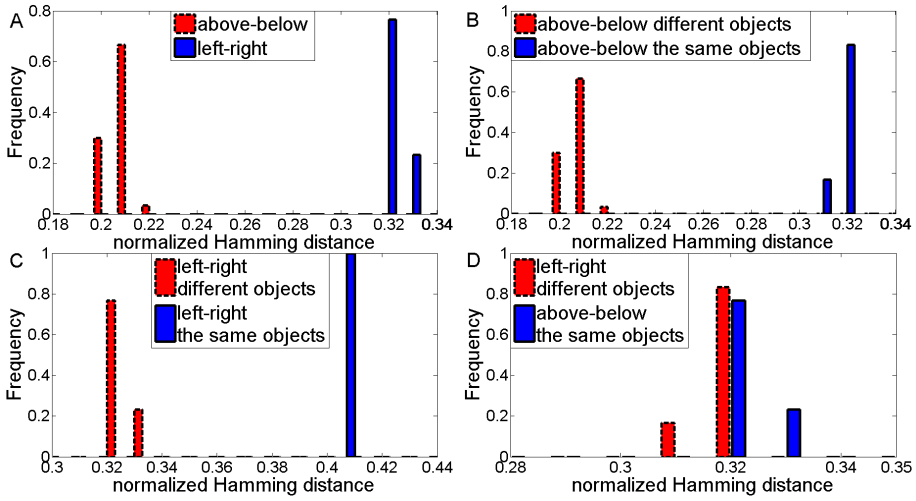

Figure 6: The results of simulations of transfer tests. Results are presented in the form histograms for 30 trials displaying the frequency of normalized Hamming distances to sucrose reward for presented alternatives. In all simulations of transfer tests the system learned the above-below concept. **A**. Simulation of transfer test 1. Distribution of distances indicates preference to stimuli with the above-below concept. The results are in correspondence with transfer test 1 in Figure 4. Mean normalized Hamming distance for above-below trials: 0.2077 ± 0.0039; for left-right trials: 0.3219 ± 0.0037. **B**. Simulation of transfer test 2. Distribution of distances indicates preference to stimuli with different objects. The results are in correspondence with transfer test 2 in Figure 4. Mean normalized Hamming distance for trials with different objects: 0.2071 ± 0.0039; for trials with the same objects: 0.3186 ± 0.0032. **C**. Simulation of transfer test 3. Distribution of distances indicates preference to stimuli with different objects. The results are in correspondence with transfer test 3 in Figure 4. Mean normalized Hamming distance for trials with different objects: 0.3219 ± 0.0037; for trials with the same objects: 0.4137 ± 0.0014. **D**. Simulation of transfer test 4. Distribution of distances of two stimuli are extremely close, i.e. there is no preference to any of stimuli. The results are in correspondence with transfer test 4 in Figure 4. Mean normalized Hamming distance for left-right trials with different objects: 0.3219 ± 0.0037; for above-below trials with the same objects: 0.3186 ± 0.0032.

the testing phase the learning was turned off.

6.1. Results of transfer tests simulation

Four simulations as described above were conducted repeating the four types of transfer tests described in Section 3.1.

The results of the four transfer tests simulations are presented in Figure 6. The figure show normalized distributions of normalized Hamming distances from $\mathbf{REWARD}_1^*$ and $\mathbf{REWARD}_2^*$ to **REWARD** for 30 different testing trials. The obtained results are positive. The concept recall accuracy in all four simulations resembles the results of the real-world experiments very closely. In particular, in transfer tests one, two and three the system recognized the correct concept in all the test trials which is align with the real-world experiment. The simulation for the fourth transfer test provides interesting results. Distributions of normalized Hamming distances to **REWARD** are very close and intersect in this test. Thus, random variations of normalized Hamming distances determine which stimulus is chosen. This behavior is equivalent to random choice between stimuli in the real-world experiments.

7. Discussion and Conclusion

This article presented an artificial online concept learning system, which imitates a part of the cognitive behavior of a living organism - the concept learning by honey bees. The system adopts Vector Symbolic Architectures for implementing the representation of a episode, the associative memory and the reasoning. The proposed design for the online learning system is generic in a sense that it allows reasoning on any relations, which are discovered at the system's runtime. The main achievements of the article are summarized as:

- Vector symbolic architectures was used to represent the structured combination of features/relations and applying those relations to novel structures in the online learning context;
- The choice of the relation represesentation is motivated by the classical analogical similarity models;
- VSA representation of relations forms a central part of the learning pipeline with a perceptual circuitry and makes feasible applying those relations to novel structures without requiring the massive training that is needed for classical neural networks;
- A proof-of-concept validation of the proposed online learning pipeline showcased on the example of the discovered concept learning capabilities by honey bees.

The proposed model is definitely biologically inspired. While its biological plausibility is, of course, a subject for future investigation, the choice of the VSA format for relation representation described in this work is tailored to the specifics of the particular real-world experiment with living organism. That is the experiments concerned demonstration of honey bees' capabilities to recognize the properties of a pair of objects as well as the relations in which they are engaged. The question whether this is a true limitation of the cognitive capabilities of bees' brain requires additional studies. However, for the considered "level of cognition" the presented system imitates the real-life scenario very well. This is a very positive result of this work and is a definite step towards closing the gap identified in Sandin et al. (2014) between VSA, small brains and neuromorphic computing.

As for the organisms with the higher cognitive capabilities (a system which is able to analyze episodes with more than two objects), more research is needed to design the VSA templates for online creation of VSA encoding of the relations between multiple objects. While this questions is the subject for the ongoing work as part of the discussion it is worth noting that this challenge is feasible to address due to discovered bounds on the number of objects, which higher animals may simultaneously focus on. This means that the number of templates for the analysis of even very complex episodes is finite.

This article focused on the proof-of-concept validation of the applicability of VSA for imitating the learning behavior of the living organisms. The main claim of this

article is that there is a class of simple artificial systems, which do not require implementation of computationally intensive cognitive architectures, e.g. based on the usage of Sparse Distributed Memory Kanerva (1988). In certain applications rather advanced cognitive behavior could be implemented using simple techniques.

Another aspect, which is currently under investigation is the time performance of the artificial online learning system. Specifically it is interesting to investigate different approaches for reward collection and the timing for re-training the system to recognize new concepts. On the implementation side the proposed architecture has very promising performance properties since all reasoning is done by simple arithmetic operations on binary codewords.

In this work traditional image processing algorithms were considered for the implementation of the vision circuitry. Also the episode content was static, i.e. did not change in time. There is definitely a potential for more bio-plausible solutions like neuromorphic vision systems Sandin et al. (2014). For modeling higher level cognition it is important for the system to be able to capture motions and the temporal context as objects in the episode analysis. Related challenge is an encoding problem, when considering episode with real-world objects. A possible way to include episode dynamics in the analysis is the usage of a long-term memory model for sequence storage, namely Sparse Distributed Memory (SDM) Kanerva (1988) or other similar approaches, e.g. Rutledge-Taylor et al. (2014).

Acknowledgements

This work is partially supported by the Swedish Foundation for International Cooperation in Research and Higher Education (STINT), institutional grant IG2011-2025.

References

Aerts, D., Czachor, M., De Moor, B., 2009. Geometric analogue of holographic reduced representation. Journal of Mathematical Psychology 53, 389-398.

Avarguès-Weber, A., de Brito Sanchez, M. G., Giurfa, M., Dyer, A. G., 2010. Aversive reinforcement improves visual discrimination learning in free flying honeybees. PLoS ONE 5 (10), 1-11.

Avarguès-Weber, A., Dyer, A. G., Giurfa, M., 2011. Conceptualization of above and below relationships by an insect. Proceedings of the Royal Society B: Biological Sciences 278, 898-905.

Avarguès-Weber, A., Dyer, A. G., Combe, M., Giurfa, M., 2012. Simultaneous mastering of two abstract concepts with a miniature brain. Proceedings of the National Academy of Sciences 109, 7481-7486.

Avarguès-Weber, A., Giurfa, M., 2013. Conceptual learning by miniature brains. Proceedings of the Royal Society B: Biological Sciences 280, 1-9.

Avarguès-Weber, A., d'Amaro, D., Metzler, M., Dyer, A. G., 2014. Conceptualization of relative size by honeybees. Frontiers in Behavioral Neuroscience 8, 1-8.

Avarguès-Weber, A., Dyer, A. G., Ferrah, N., Giurfa, M., 2015. The forest or the trees: preference for global over local image processing is reversed by prior experience in honeybees. Proceedings of the Royal Society B: Biological Sciences 282.

Dyer, A. G., Ravi, S., Garcia, J. E., 2014. Flying in Complex Environments: Can Insects Bind Multiple Sensory Perceptions and What Could Be the Lessons for Machine Vision? Journal of Software Engineering and Applications 7, 406-412.

Eliasmith, C., 2013. How to Build a Brain: A Neural Architecture for Biological Cognition. Oxford University Press.

Emruli, B., Gayler, R. W., Sandin, F., 2013. Analogical mapping and inference with binary spatter codes and sparse distributed memory. In: The proceedings of the international joint conference on neural networks, pp. 1-8.

Emruli, B., Sandin, F., 2014. Analogical mapping with sparse distributed memory: A simple model that learns to generalize from examples. Cognitive Computation 6 (1), 74-88.

Emruli, B., Sandin, F., Delsing, J., 2014. Vector space architecture for emergent interoperability of systems by learning from demonstration. Biologically Inspired Cognitive Architectures 9, 33-45.

Gallant, S. I., Okaywe, T. W., 2013. Representing objects, relations, and sequences. Neural Computation 25 (8), 2038-2078.

Gayler, R. W., 1998. Multiplicative binding, representation operators & analogy. In: Gentner, D., Holyoak, K. J., Kokinov, B. N. (Eds.), Advances in analogy research: Integration of theory and data from the cognitive, computational, and neural sciences. New Bulgarian University, Sofia, Bulgaria, pp. 1-4.

Gayler, R. W., Levy, S. D., 2009. A distributed basis for analogical mapping. In: The Proceeding of the second International Conference on Analogy, 2009. pp. 1-10.

Gentner, D., 1983. Structure-mapping: A theoretical framework for analogy. Cognitive Science, 7, 155-170.

Giurfa, M., Zhang, S., Jenett, A., Menzel, R., Srinivasan, M. V., 2001. The concepts of "Sameness" and "Difference" in an insect. Nature 410, 930-933.

Kanerva, P., 1988. Sparse Distributed Memory. The MIT Press.

Kanerva, P., 1997. Fully distributed representation. In: Real World Computing Symposium. Vol. 97. pp. 358-365.

Kanerva, P. 2000. Large patterns make great symbols: An example of learning from example. In: Stefan Wermter and R. Sun (Eds.). Hybrid neural systems, Vol. 1778, pp. 194-203.

Kanerva, P., 2009. Hyperdimensional computing: an introduction to computing in distributed representation with high-dimensional random vectors. Cognitive Computation 1 (2), 139-159.

Kleyko, D., Osipov, E., 2014. Brain-like classifier of temporal patterns. In: The Proceeding of the 2nd International Conference on Computer and Information Sciences - ICCOINS, 2014. pp. 1-6.

Kleyko, D., Osipov, E., 2014. On Bidirectional Transitions between Localist and Distributed Representations: The Case of Common Substrings Search Using Vector Symbolic Architecture. Procedia Computer Science 41, 104-113.

Kleyko, D., Osipov, E., Senior, A., Khan, A. I., Sekercioglu Y. A., 2015. Holographic Graph Neuron: a Bio-Inspired Architecture for Pattern Processing. Preprint at http://arxiv.org/pdf/1401.0742v1.pdf.

Kleyko, D., Osipov, E., Papakonstantinou, N., Vyatkin, V., Mousavi, A., 2015. Fault Detection in the Hyperspace: Towards Intelligent Automation Systems. In: The Proceeding of the 13th IEEE International Conference on Industrial Informatics - INDIN, 2015. pp. 1-6.

Levy, S. D., Bajracharya, S., Gayler, R. W., 2013. Learning Behavior Hierarchies via High-Dimensional Sensor Projection. In: The Proceeding of the Twenty-Seventh AAAI Conference on Artificial Intelligence, 2013.

Levy, S. D., Lowney, C., Meroney, W., Gayler, R. W., 2014. Bracketing the Beetle: How Wittgenstein's Understanding of Language Can Guide Our Practice in AGI and Cognitive Science. In: Artificial General Intelligence, Lecture Notes in Computer Science Vol. 8598, 73-84.

Markman, A., Gentner, D., 2005. Nonintentional similarity processing. The New Unconscious. Oxford University Press. 107–137.

Miller, G. A., 1956. The magical number seven, plus or minus two: Some limits on our capacity for processing information. Psychological Review 63 (2), 81-97.

Newell, A., 1994. Unified Theories of Cognition. Harvard University Press.

Nosofsky, R., 1986. Attention, similarity and the identification-categorization relationship. Journal of Experimental Psychology: General, 115(1), 39-57.

Plate, T. A., 2003. Holographic reduced representations: Distributed representation for cognitive structures. Center for the Study of Language and Information (CSLI).

Rachkovskij, D. A., Kussul, E. M., 2001. Binding and normalization of binary sparse distributed representations by context-dependent thinning. Neural Computation 13 (2), 411-452.

Rachkovskij, D. A., Kussul, E. M., Baidyk, T.N., 2013. Building a world model with structure-sensitive sparse binary distributed representations. Biologically Inspired Cognitive Architectures 3, 64-86.

Rasmussen, D., Eliasmith, C., 2011. A Neural Model of Rule Generation in Inductive Reasoning. Topics in Cognitive Science 3 (1), 140-153.

Rutledge-Taylor, M. F., Kelly, M. A., West, R. L., Pyke, A. A., 2014. Dynamically structured holographic memory. Biologically Inspired Cognitive Architectures 9, 9-32.

Räsänen, O., Kakouros, S., 2014. Modeling Dependencies in Multiple Parallel Data Streams with Hyperdimensional Computing. IEEE Signal Processing Letters 21 (7), 899-903.

Sandin, F., Khan, A. I., Dyer, A. G., Amin, A. H. M., Indiveri, G., Chicca, E., Osipov, E., 2014. Concept Learning in Neuromorphic Vision Systems: What Can We Learn from Insects? Journal of Software Engineering and Applications 7, 387-395.

Shepard, R., 1962. The analysis of proximities: Multidimensional scaling with an unknown distance function, I. Psychometrika, 27(2), 125-140.

Snaider, J., Franklin, S., 2014. Modular Composite Representation. Cognitive Computation, 6(3), 510-527.

Soccol, D., Thurrowgood, S., Srinivasan, M. V., 2007. A vision system for optic-flow-based guidance of UAVs. In: Proceedings of the Ninth Australasian Conference on Robotics and Automation, 2007. pp. 1-7.

Srinivasan, M. V., 2006. Small brains, smart computations: Vision and navigation in honeybees, and applications to robotics. In: International Congress Series: 2nd International Conference on Brain-inspired Information Technology 1291, 30-37.

Srinivasan, M. V., 2010. Honey bees as a model for vision, perception, and cognition. Annual review of entomology 55, 267-284.

Srinivasan, M. V., 2011. Visual control of navigation in insects and its relevance for robotics. Current opinion in neurobiology 21, 535-543.

Srinivasan, M. V., Zhang, S., Altwein, M. and Tautz, J., 2000. Honeybee navigation: nature and calibration of the ÊżodometerÊij. Science 287, 851-853.

Tversky, A., Kahneman, D., 1986. Rational choice and the framing of decisions. Journal of Business, 59(4), S251-S278.